Arquitectura de computadoras
Smile 2.0 : Manual de usuario
Trabajo práctico especial

Lata Andrea 48102
Goldberg Daniel 48421
Kenny Kevin 49262

18 de junio del 2010

Índice

1. Introducción

El objetivo de este manual es enseñar al usuario de Smile cómo se utiliza para que pueda explotarlo al máximo.

2. Smile

2.1. Comandos

2.1.1. Comandos básicos

- *ayuda*: imprime en pantalla los posibles comandos de ayuda que se pueden ingresar

```
<Smile2.0>: ayuda
AYUDA
Comandos validos:
.        ayuda exc
.        ayuda gdt
.        ayuda mem
.        ayuda cpuid
.        ayuda printf
.        ayuda scanf
```

- *ayuda exc*: imprime en pantalla un menú con las posibles pruebas de excepciones que puede ejecutar el usuario

```
<Smile2.0>:ayuda exc
AYUDA DE EXCEPCIONES
Comandos validos:
.        Divide
.        Overflow
.        GralProtection
```

- *ayuda gdt*: imprime en pantalla una breve descripción del comando gdt

```
<Smile2.0>:ayuda gdt
AYUDA GDT
.        gdt: Con el comando "gdt" podra crear un segmento en la tabla de
descriptores global. El menu lo ira guiando, usted podra elegir el Indice
del mismo, el Tipo (Codigo o Dato), los privilegios, la direccion de
memoria base y la limite.
.        infogdt
.        infodescriptor
```

- *ayuda mem*: imprime en pantalla un menú con los comandos que puede utilizar la memoria

```
<Smile2.0>:ayuda mem
AYUDA MEMORIA
.        escribirmem
.        leermem
```

- *ayuda cpuid*: imprime en pantalla la funcionalidad del comando cpuid

```
<Smile2.0>: ayuda cpuid
AYUDA CPUID
El comando "cpuid" nos brinda informacion sobre la PC donde estamos trabajando.
Muestra la lista de los "features" soportados; para obtener informacion sobre uno
en particular utilizar el comando "infofeature"
```

- *ayuda printf*: imprime en pantalla la funcionalidad del comando printf

```
<Smile2.0>: ayuda printf
AYUDA PRINTF
El comando "printf" nos permite testear la funcion printF, pidiendonos nuestro nombre
y nuestra edad, concatenandolos y devolviendonos la informacion correspondiente
```

- *ayuda scanf*: imprime en pantalla la funcionalidad del comando scanf

```
<Smile2.0>: ayuda scanf
AYUDA SCANF
El comando "scanf" nos permite testear la funcion scanF, pidiendonos un entero,
un entero y una cadena concatenados, los separa y devuelve la informacion
correspondiente
```

2.1.2. Comandos excepciones

- *divide*: le pide al usuario que ingrese un dividendo y un divisor, y ejecuta la división

```
<Smile2.0>: divide
Por favor ingrese el dividendo:
Por favor ingrese le divisor:
```

- *overflow*: genera una excepción de overflow

```
<Smile2.0>:overflow
Se ha producido un overflow
```

- *GralProteccion*: genera una excepción de Error de proteccion general

```
<Smile2.0>:GralProteccion
Error de proteccion general
```

2.1.3. Comandos gdt

- *gdt*: podrá crear un segmento en la tabla de descriptores global

```
<Smile2.0>:gdt
Ingrese el indice donde desea ubicar el segmento:

El segmento que desea agregar es de tipo:
1. De sistema: puerta de llamada, TSS, tec.
2. Normal: codigo datos o pila.
```

```
Ingrese su eleccion ( 1 - 2):

Indique el nivel de privilegio del segmento:
1. Privilegio nivel 0 (Maximo).
2. Privilegio nivel 1.
3. Privilegio nivel 2.
4. Privilegio nivel 3 (Minimo).
Ingrese su eleccion ( 1 - 4):

Indique el Tipo del segmento:
1. De codigo.
2. De dato.
Ingrese su eleccion ( 1 - 2):

Ingrese Direccion base en Hexadecimal:

Ingrese Direccion Limite en Hexadecimal:
```

- *infogdt*: podrá imprimir la información de todos los descriptores presentes en la gdt

```
<Smile2.0>:infogdt
TABLA DE DESCRIPTORES GLOBAL:
gdt[0x1]=Code segment, linearadd = 0, limit=FFFF * 4K bytes, Execute/Read
gdt[0x2]=Data segment, linearadd = 0, limit=FFFF * 4K bytes, Read/Write, Accessed
gdt[0x3]=Code segment, linearadd = 0, limit=FFFF bytes, Execute/Read, Conforming
gdt[0x4]=Data segment, linearadd = 0, limit=FFFF bytes, Read/Write, Accessed
gdt[0x5]=Data segment, linearadd = B8000, limit=FFFF bytes, Read/Write
```

- *infodescriptor*: podrá imprimir la información de un descriptor de la gdt

```
<Smile2.0>: infodescriptor
Ingrese indice del descriptor en Hexadecimal: a
DESCRIPTOR:
gdt[0xA] = Data segment, linearadd=0, limit=0 bytes, Read/Write
```

ejemplo: el usuario ingresó una 'a'

2.1.4. Comandos memoria

- *escribirmem*: podrá escribir en un segmento de la gdt

```
<Smile2.0>:escribirmem
Ingrese el indice del segmento donde desea escribir:
Ingrese el texto que desea agregar:
Ingrese desplazamiento:
```

- *leermem*: podrá leer un segmento de la gdt

```
<Smile2.0>:leermem
Ingrese el indice del segmento que desea leer:
Ingrese desplazamiento:
Ingrese la longitud:
```

2.1.5. Comandos cpuid

- *cpuid*: imprime la informacion sobre la CPU

```
<Smile2.0>: cpuid
ID FABRICANTE:

Features soportadas:
```

- *infofeature*: devuelve informacion basica sobre un feature

```
<Smile2.0>: infofeature
Ingrese feature(1-29):
```

2.1.6. Comandos stdio

- *printf*: muestra un ejemplo de uso de printf

```
<Smile2.0>: printf
Ingrese su nombre:

Ingrese su edad:

Usted se llama    y tiene    anios
```

- *scanf*: muestra un ejemplo de uso de scanf

```
<Smile2.0>: scanf
Ingrese un entero, un caracter y un string

ingreso:
un entero:
un caracter:
una cadena:
```

2.2. Dispositivos soportados

- teclado

2.2.1. Funcionalidad de los dispositivos soportados

El teclado se encuentra en idioma inglés. Cualquier caracter que no pertenezca a este lenguaje, no se puede asegurar que será soportado. Los caracteres soportados son las letras, los números, el backspace, el enter, el Rshift, el Lshift, los símbolos de puntuación y el espacio.

2.3. Forma de ejecutar

1. Colocar el diskette o cd
2. Encender la PC y hacerla bootear por este medio.
3. En la pantalla podrá ver Kernel de prueba

4. Debe ingresar enter sobre Kernel
5. Ya podrá acceder a ingresar comandos en linea de comando